**Name : Ali Azhar**

**Roll No : SU92-BSAIM-S24-026**

**Section : 2A**

**Project Report**

**1. Overview**

The **Employee Management System** is a Python-based project designed to manage employee records efficiently. It provides features for adding, displaying, updating, deleting, and promoting employees. The system supports two types of employees:

* **Managers**: Employees with department details.
* **Workers**: Employees with details about the hours worked.

All employee data is stored in a **CSV file** named Management.csv, ensuring persistent storage.

**2. Key Features**

1. **Login System**:
   * Ensures only authorized users can access the system.
   * Predefined credentials: Username: Laptop, Password: Project2024.
2. **Employee Types**:
   * **Manager**: Has attributes like name, age, salary, and department.
   * **Worker**: Has attributes like name, age, salary, and hours worked.
3. **Functionalities**:
   * **Add Employee**: Allows adding new Managers or Workers with relevant details.
   * **Display All Employees**: Reads and displays employee information from the CSV file in a user-friendly format.
   * **Update Employee**: Updates employee details such as name, age, salary, department (for Managers), or hours worked (for Workers).
   * **Delete Employee**: Deletes an employee's record based on their name.
   * **Promote Employee**: Increases an employee’s salary by a specified percentage.

**3. Implementation Details**

1. **Classes and Inheritance**:
   * **Employee (Base Class)**:
     + Attributes: name, age, and salary (private).
     + Getter and setter methods for encapsulation.
   * **Manager (Derived Class)**:
     + Inherits from Employee.
     + Adds the dept attribute for the department.
   * **Worker (Derived Class)**:
     + Inherits from Employee.
     + Adds the hours\_worked attribute.
2. **File Handling**:
   * Data is stored in the Management.csv file.
   * The csv.DictReader and csv.DictWriter modules are used for reading and writing employee data in CSV format.
3. **Functional Modularity**:
   * Separate functions for each operation (add, display, update, delete, promote) to ensure code clarity and maintainability.

**4. User Flo**

1. **Login**:
   * Users must log in with valid credentials.
2. **Menu**:
   * After login, the user accesses a menu to select from available operations.
3. **Operations**:
   * Depending on the choice, the system performs the selected operation and updates the CSV file accordingly.

**5. Advantages**

1. **Simple and Interactive**: Provides an easy-to-navigate text-based interface.
2. **Persistent Storage**: Employee data is saved in a CSV file, ensuring it is retained even after the program ends.
3. **Scalable**: New employee types or additional features can be integrated with minimal changes.
4. **Encapsulation**: Uses getter and setter methods for secure access to employee attributes.

**6. Limitations**

1. **Hardcoded Credentials**: The login system uses a fixed username and password, which could be a security concern.
2. **No Data Validation**: Inputs like age, salary, and hours worked are not validated for negative values or incorrect formats.
3. **Single CSV File**: Using a CSV file limits the performance and scalability compared to a database.

**7. Future Enhancements**

1. **Dynamic Credentials**: Add user management with the ability to create and update login credentials.
2. **Database Integration**: Replace CSV file storage with a database for enhanced performance and scalability.
3. **Improved Validation**: Implement robust input validation to prevent incorrect data entries.
4. **GUI Implementation**: Develop a graphical user interface for better user experience.

**8. Conclusion**

This project demonstrates the use of **object-oriented programming**, **file handling**, and **inheritance** in Python. It is an efficient solution for small-scale employee management needs. With additional features and enhancements, it has the potential to evolve into a more robust and scalable system.